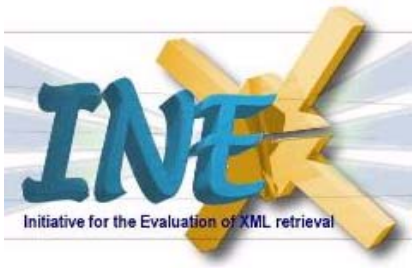

INEX'03 Retrieval Task and Result Submission Format Specification

Retrieval Task

The retrieval task to be performed by the participating groups of INEX'03 is defined as the ad-hoc retrieval of XML documents. In information retrieval literature, ad-hoc retrieval is described as a simulation of how a library might be used, and it involves the searching of a static set of documents using a new set of topics. While the principle is the same, the difference for INEX is that the library consists of XML documents, the queries may contain both content and structural conditions and, in response to a query, arbitrary XML elements may be retrieved from the library. Within the ad-hoc retrieval task we define the following three sub-tasks:

CO: Content-oriented XML retrieval using content-only (CO) queries. As described in the INEX'03 Topic Development Guide, CO queries are requests that ignore the document structure and contain only content related conditions, e.g. only specify what a document/component should be about (without specifying what that component is). The need for this type of query for the evaluation of XML retrieval stems from the fact that users may not care about the structure of the result components or may not be familiar with the exact structure of the XML documents. In this task, it is left to the retrieval system to identify the most appropriate XML elements to return to the user. These elements are components that are most specific and most exhaustive with respect to the topic of request. Most specific here means that the component is highly focused on the topic, while exhaustive reflects that the topic is exhaustively discussed within the component.

SCAS: Content-oriented XML retrieval based on content-and-structure (CAS) queries, where the structural constraints of a query must be strictly matched. CAS queries are topic statements, which contain explicit references to the XML structure, and explicitly specify the contexts of the user's interest (e.g. target elements) and/or the contexts of certain search concepts (e.g. containment conditions). In this task, the user's query is considered as an exact formulation of his/her information need, where the structural conditions specified within the query must be satisfied exactly by the retrieved components.

VCAS: Content-oriented XML retrieval based on content-and-structure (CAS) queries, where the structural constraints of a query can be treated as vague conditions. This task deviates from the previous one in that XML elements 'structurally similar' to those specified in the query may be considered correct answers. The idea behind this sub-task is to allow the evaluation of XML retrieval systems that aim to implement a more fuzzy approach to XML retrieval, where not only the content conditions within a user query are treated with uncertainty but also the expressed structural conditions. These systems aim to return components that contain the information sought after by the user even if the result elements do not exactly meet the structural conditions expressed in the query.

The actual search queries put to the retrieval engines (e.g. used to search the document collection) may be generated either manually or automatically from any part of the topics, with the exception of the narrative. Please note that at least one submitted run for each sub-task must be with the use of automatic queries.

Result Submission

For each sub-task up to 3 runs may be submitted. The results of one run must be contained in one submission file (e.g. up to 9 files can be submitted in total). A submission may contain up to **1500** retrieval results for each of the INEX topics included within that sub-task (e.g. for the CO sub-task only submit the search results obtained for the CO topics).

Submission format

For relevance assessments and the evaluation of the results we require submission files to be in the format described in this section. The overall submission format is defined in the following DTD:

```
<!ELEMENT inex-submission          (description, topic+)>
<!ATTLIST inex-submission
     participant-id          CDATA         #REQUIRED
     run-id                  CDATA         #REQUIRED
     task     ( CO | SCAS | VCAS )         #REQUIRED
     query    (automatic | manual)         #REQUIRED
     topic-part  (T | D | K | TD | TK | DK | TDK)   #REQUIRED
>
<!ELEMENT description         (#PCDATA)>
<!ELEMENT topic               (result*)>
<!ATTLIST topic
     topic-id                CDATA         #REQUIRED
>
<!ELEMENT result              (file, path, rank?, rsv?)>
<!ELEMENT file                (#PCDATA)>
<!ELEMENT path                (#PCDATA)>
<!ELEMENT rank                (#PCDATA)>
<!ELEMENT rsv                 (#PCDATA)>
```

Each submission must specify the following information:

- `participant-id`: the participant ID of the submitting institute (available at http://inex.is.informatik.uni-duisburg.de:2003/inex03/servlet/ShowParticipants),
- `run-id`: a run ID (which must be unique for the submissions sent from one organisation – also please use meaningful names as much as possible),
- `task`: the identification of the task (e.g. `CO`, `SCAS` or `VCAS`),
- `query`: the identification of whether the query was constructed automatically or manually from the topic,
- `topic-part`: the specification of whether the automatic or manual query was generated from the topic title only (`T`), the topic description only (`D`), the keywords only (`K`), the combination of the topic title and the topic description (`TD`), the combination of the topic title and the keywords (`TK`), the combination of the topic description and keywords (`DK`), or the combination of the topic title, topic description and keywords (`TDK`).

Furthermore each submitted run must contain a (brief) `description` of the retrieval approach applied to generate the search results.

A submission should then contain a number of `topic`s, each identified by its topic ID (as provided with the topics). For each topic a maximum of **1500** `result` elements may be included. A result element is described by a `file` name and an element `path` and it may include `rank` and/or retrieval status value (`rsv`) information.

Before detailing these elements, below is a sample submission file:

```
<inex-submission participant-id="12" run-id="VSM_Aggr_06" task="CO"
query="automatic" topic-part="TK">
    <description>Using VSM to compute RSV at leaf level combined with
          aggregation at retrieval time, assuming independence and using
          acc=0.6.
    </description>
```

```
    <topic topic-id="01">
        <result>
            <file>tc/2001/t0111</file>
             <path>/article[1]/bm[1]/ack[1]</path>
            <rsv>0.67</rsv>
        </result>
        <result>
            <file>an/1995/a1004</file>
            <path>/article[1]/bdy[1]/sec[1]/p[3]</path>
            <rsv>0.1</rsv>
        </result>
        [ ... ]
    </topic>
    <topic topic-id="02">
        [ ... ]
    </topic>
    [ ... ]
</inex-submission>
```

Rank and RSV

The `rank` and `rsv` elements are provided for submissions based on a retrieval approach producing ranked output. The ranking of the result elements can be described in terms of

- Rank values, which are consecutive natural numbers, starting with 1. Note that there can be more than one element per rank.
- Retrieval status values (RSVs), which are positive real numbers. Note that there may be several elements having the same RSV value.

Either of these methods may be used to describe the ranking within a submission. If both rank and rsv are given, the rank value is used for evaluation. These elements may be omitted from a submission if a retrieval approach does not produce ranked output.

File and path

Since XML retrieval approaches may return arbitrary XML nodes from the documents of the INEX collection, we need a way to identify these nodes without ambiguity. Within INEX submissions, elements are identified by means of a `file` name and an element (node) `path` specification, which must be given in XPath syntax.

File names must be given relative to the INEX collection's "xml" directory (excluding the "xml" directory itself from the file path). The file path should use '/' for separating directories. Note that only article files (e.g. no "volume.xml" files) can be referenced here. The extension ".xml" must be left out. Example:

```
an/1995/a1004
```

Element paths are given in XPath syntax. To be more precise, only fully specified paths are allowed, as described by the following grammar:

```
Path          ::= '/' ElementNode Path
                | '/' ElementNode '/' AttributeNode
                | '/' ElementNode
ElementNode   ::= ElementName Index
AttributeNode ::= '@' AttributeName
Index         ::= '[' integer ']'
```

Example:

```
/article[1]/bdy[1]/sec[4]/p[3]
```

This path identifies the element which can be found if we start at the document root, select the first "article" element, then within that, select the first "bdy" element, within which we select the fourth "sec" element, and finally within that element we select the third "p" element. Note that XPath counts

elements starting with 1 and takes into account the element type, e.g. if a section had a title and two paragraphs then their paths would be given as: `../title[1], ../p[1]` and `../p[2]`.

When producing the XPath expressions of result elements, the equivalent-tags rules (see INEX'03 Guidelines for Topic Development) must be ignored, e.g. result elements must be identified in line with the original structure of the XML documents! For example, given the structure: `<sec><p>..</p><ip5>..</ip5><p>..</p></sec>)` the following XPaths should be generated: `/sec[1], /sec[1]/p[1], /sec[1]/ip5[1],` and `/sec[1]/p[2]`. Note that the same structure, taking into account the equivalent-tags rules, would result in the XPaths: `/sec[1], /sec[1]/p[1], /sec[1]/p[2],` and `/sec[1]/p[3]`. However, result elements identified by the latter XPaths will lead to incorrect evaluations of the submitted runs.

A result element is identified unambiguously using the combination of its file name and element path. Example:

```
<result>
    <file>an/1995/a1004</file>
    <path>/article[1]/bdy[1]/sec[1]/p[3]</path>
</result>
```

An application that can be used to check the correctness of a given path specification is available at
http://inex.is.informatik.uni-duisburg.de:2003/browse.html
Note that this application requires the input of a file name and element path. If these are correctly given, the specified XML element within its container article element will be displayed.

Result Submission Procedure

An online submission tool will be provided. Details on how to submit will be circulated as part of a separate document in the near future.

July 23, 2003
Gabriella Kazai, Mounia Lalmas, Norbert Goevert and Saadia Malik